Efficient CUR Matrix Decomposition via Relative-Error Double-Sided Least Squares Solving

Qi Luan
Ph.D. Program in Mathematics
The City University of New York, Graduate Center
qluan@gradcenter.cuny.edu

Liang Zhao
Department of Computer Science
The City University of New York, Lehman College
Liang.Zhao1@lehman.cuny.edu

***Abstract*—Matrix CUR decomposition aims at representing a large matrix A with the product $C \cdot U \cdot R$, where C (resp. R) consists of a small collection of the original columns (resp. rows), and U is a small intermediate matrix connecting C and R. While modern randomized CUR algorithms have provided many efficient methods of choosing representative columns and rows, there hasn't been a method to find the optimal U matrix. In this paper, we present a sublinear-time randomized method to find good choices of the U matrix. Our proposed algorithm treats the task of finding U as a double-sided least squares problem $\min_Z \|A - CZR\|_F$, and is able to guarantee a close-to-optimal solution by solving a down-sampled problem of much smaller size. We provide worst-case analysis on its approximation error relative to theoretical optimal low-rank approximation error, and we demonstrate empirically how this method can improve the approximation of several large-scale real data matrices with a small number of additional computations.**

***Index Terms*—Matrix Decomposition, Sublinear Algorithm, Stochastic Algorithm, CUR Decomposition.**

I. Introduction

Compressing a large matrix is a common task in statistical learning, fueled by the explosive growth in the size of modern datasets. A common practice is to approximate the original matrix using a low-rank matrix, represented as a product of several matrices of comparatively much smaller size. While the truncated Singular Value Decomposition (SVD) is capable of producing such a rank-k approximation with minimal norm on the approximation error, the complexity of deterministic SVD algorithms is super-linear in the size of the data, and the factorization obtained from it consists of linear combinations of many rows or columns, which often bear little concrete meaning and are impossible to interpret.

Matrix CUR decomposition aims to find low-rank matrix approximation with original matrix rows and columns. More precisely, such a factorization of an $m \times n$ matrix A is represented as $C \cdot U \cdot R$, where C consists of d_1 columns of A, and R consists of d_2 rows of A, where $d_1, d_2 \ll \min(m, n)$. To effectively approximate A, the product CUR is required to have approximation error close to the minimal error, and the algorithm that computes the approximation must be significantly faster than the SVD method, preferably using linear time in input sparsity. Both goals have been asymptotically achieved by sampling-based randomized CUR algorithms [1], [3], [9] with controllable failure probabilities. A common pattern of these algorithms is to first construct C as a small subset of columns sampled with a probability distribution reflecting the "importance" of each column, then sample rows to obtain R, and lastly construct an appropriate middle factor U connecting C and R. Up to constant factors, the error norm and computational complexity are optimal in the number of columns and rows being sampled. This approach has been shown to be highly effective in the principle component analysis of large datasets [1].

This paper develops a method that can be used to further improve existing sampling-based CUR algorithms by providing a near-optimal choice of the middle block U. Given factor C and R, we treat the task of finding U as *double-sided least squares* problem $\min_Z \|A - CZR\|_F$. The optimal solution is $Z_{opt} = C^+ A R^+$, where M^+ represents the Moore-Penrose matrix pseudo inverse. Ideally one would use $U = Z_{opt}$ to form a CUR approximation, but its cost is unbearable when the size of A is much greater than the size of Z. We instead develop a randomized algorithm that solves a down-sampled problem

$$\min_{Z \in \mathbb{R}^{d_1 \times d_2}} \sum_{(i,j) \in S} \frac{(A_{ij} - C_i Z R^j)^2}{p_{ij}|S|}. \tag{1}$$

Here the index set S is a small subset of matrix indicies sampled with replacement according to probability distribution $\{p_{ij}\}_{1 \le i \le m, 1 \le j \le n}$. Problem (1) is much easier to solve than the double-sided least squares problem. We show that if the sampling probabilities are carefully chosen, its solution can well approximate Z_{opt}, and thus providing a better overall CUR approximation to the matrix A.

We run numerical tests on several large-scale matrices representing real-world datasets, some of which contain over one billion values. Combined with a sampling strategy that uses leverage-scores on general matrices [3] or uniform-sampling on partially observed low-coherence matrices [11], our algorithm can produce CUR approximations with approximation error constantly closer to the optimal error comparing to existing CUR algorithms.

This work was supported by NSF Grants CCF–1733834 and PSC CUNY Award 63749-00 51.

2375-0197/20/$31.00 ©2020 IEEE
DOI 10.1109/ICTAI50040.2020.00056

II. Related Work

The study of CUR matrix approximation problem can be traced back to pseudo-skeleton decomposition [4], [5], [7], where algorithms have been developed in conjunction with the study of finding maximal volume matrix sub-blocks. This problem has been widely investigated in both Theoretical Computer Science community and Numerical Linear Algebra community. Drineas, Kannan, and Mahoney [2] propose a randomized CUR algorithm with additive error and $O(m+n)$ space and time. Drineas, Mahoney, and Muthukrishnan [3] propose a sampling CUR algorithm that achieves relative error using $O(k\log(k)\epsilon^{-2})$ rows and columns. Wang and Zhang [9] develop an adaptive sampling method that achieves relative-error CUR approximation with complexity linear in input size. Boutsidis and Woodruff [1] present an input-sparsity-time CUR algorithm using $O(k/\epsilon)$ rows and columns. We refer readers to Boutsidis and Woodruff [1] and Woodruff [10] for more complete review of the literature.

Xu, Jin, and Zhou [11] study CUR matrix approximation of low-coherence matrices that can only be observed partially, and they propose an additive-error algorithm using uniform sampling for rows/columns, as well as uniform sampling for solving the resulting double-sided least squares problem. This work differentiates from [11] in that our proposed algorithm uses sampling probabilities derived from the input matrices, and thus it is applicable to arbitrary inputs. Moreover, we show that our algorithm can achieve small relative error, which is often more desirable in practice.

III. Notations

In this paper, we use A, A^i, A_j, and $A_{i,j}$ to represent matrix, matrix column, matrix row, and matrix entry, respectively. $\|A\|_F$ is the Frobenius norm, and $\otimes$ denotes the matrix kronecker product. The singular value decomposition of $A \in \mathbb{R}^{m\times n}$ factors the matrix as a product

$$A = U\Sigma V^T \tag{2}$$

$$= \begin{bmatrix} U_k, U_k^{\perp} \end{bmatrix} \begin{bmatrix} \Sigma_k & \\ & \Sigma_{k,\perp} \end{bmatrix} \begin{bmatrix} V_k^T \\ V_k^{\perp T} \end{bmatrix}, \tag{3}$$

where (assuming $m \geq n$) $U \in \mathbb{R}^{m\times n}$ and $V \in \mathbb{R}^{n\times n}$ consist of orthonormal columns, and Σ is a diagonal matrix formed by singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$. The optimal rank-$k$ approximation A_k is formed by truncating the SVD of A, namely $A_k = U_k\Sigma_k V_k^T$, where $U_k(V_k)$ consist of the first k columns of $U(V)$, and $\Sigma_k = diag(\sigma_i)_{i\in[k]}$, where $[k] = \{1, ..., k\}$. A_k is the "best" rank-k approximation in the sense that the following equality holds:

$$\|A - A_k\|_F = \sqrt{\sum_{i=k+1}^{n} \sigma_i^2} = \min_{rank(X)=k} \left\{\|A - X\|_F\right\} \tag{4}$$

IV. Theoretical Results

In this section, we describe the proposed algorithm for double-sided least squares problem, and we present theoretical bound on its relative error as well as the probability of success.

A. Randomized Algorithm for Double-Sided Least Squares Problem

Algorithm 1 takes as input matrices $A \in \mathbb{R}^{m\times n}, C \in \mathbb{R}^{m\times d_1}$, $R \in \mathbb{R}^{d_2\times n}$ assuming $d_1, d_2 \leq \min(m, n)$, and a positive constant ϵ. The algorithm returns a matrix $Z \in \mathbb{R}^{d_1\times d_2}$ such that $CZR \approx A$, which can be interpreted as a reconstruction of A using C and R. The optimal solution to

$$\min_Z ||A - CZR||_F \tag{5}$$

is $Z_{opt} = C^+AR^+$, which requires to access the entire matrix A. However, a near-optimal solution can be obtained by using very few elements in A sampled from a non-uniform probability distribution constructed with C and R.

Specifically, Algorithm 1 first computes the top left-singular vectors U of C, V of R^T, and probability distribution

$$p_{i,j} = \frac{||U_i||_F^2||V_j||_F^2}{d_1 d_2} \text{ for } i \in [m], j \in [n]. \tag{6}$$

Then, we sample independently $c = \Theta(d_1^2 d_2^2 \epsilon^{-2})$ index pairs $\left\{(i_t, j_t)|t \in [c]\right\}$ from the probability distribution (6). According to the sampled index pairs, construct vector $Y \in \mathbb{R}^c$ and matrix $W \in \mathbb{R}^{c\times d_1 d_2}$, such that for all $t \in [c]$

$$Y_t = \frac{1}{\sqrt{cp_{i_t,j_t}}} A_{i_t,j_t} \quad \text{and} \quad W_t = \frac{1}{\sqrt{cp_{i_t,j_t}}} C_{i_t} \otimes R_{j_t}^T. \tag{7}$$

Finally, Z is computed, reshaped into a $d_1 \times d_2$ matrix

$$Z = \arg\min_x ||Y - WX||_F. \tag{8}$$

With probability of success greater or equal to 0.7 (this probability can be improved by increasing the number of the sampled index pairs), Algorithm 1 computes a near-optimal solution Z with $\|A - CZR\|_F^2 \leq (1+\epsilon)\|A - CZ_{opt}R\|_F^2$ using no more than $\Theta(d_1^2 d_2^2 \epsilon^{-2})$ elements from A, and at essentially sublinear computational cost.

Algorithm 1: Sublinear-time algorithm for approximate solution of $\min_Z \|A - CZR\|_F$

Input: $A \in \mathbb{R}^{m\times n}, C \in \mathbb{R}^{m\times d_1}, R \in \mathbb{R}^{d_2\times n}$ given $d_1, d_2 \leq \min(m, n)$, and $0 < \epsilon < 1$;
Output: $Z \in \mathbb{R}^{d_1\times d_2}$;

```
Compute probabilities p_{i,j} implicitly for i ∈ [m]
and j ∈ [n] using Eqn. (6).
c ← 3200 d_1^2 d_2^2 ε^{-2}
Initialize Y ∈ R^c and W ∈ R^{c×d_1 d_2}.
for t = 1, 2, ..., c do
    pick index pair (i_t, j_t) with probability p_{i_t,j_t}.
    Set Y_t = 1/√(c p_{i_t,j_t}) A_{i_t,j_t}
    Set W_t = 1/√(c p_{i_t,j_t}) C_{i_t} ⊗ R^T_{j_t}
end for
Solve the least squares problem
Z = arg min_X ||Y − WX||_F
return Reshaped Z ∈ R^{d_1×d_2}.
```

B. Guarantee for Algorithm 1

Before we present our analysis of Algorithm 1 in Theorem 2, we first introduce a powerful supporting theorem regarding the quality of approximation for column/row sampling.

Theorem 1. *(Adapted from Thm. 5, Alg. Exactly(c) in [3]) Let $B \in \mathbb{R}^{m\times n}$ be a matrix of rank less or equal to k, $A \in \mathbb{R}^{m\times p}$, $0 < \epsilon < 1$, and let $Z_{opt} = \arg\min_X ||A - BX||_F = B^+A$. Let U be the top k left singular vectors of B and define any probability distribution*

$$p_i \geq \frac{\beta||U_i||_F^2}{k} \text{ for all } i \in [m] \tag{9}$$

for some $0 < \beta \leq 1$. Let $c = 3200k^2\epsilon^{-2}\beta^{-1}$, $Y \in \mathbb{R}^{c\times p}$ and $W \in \mathbb{R}^{c\times n}$ be two random matrices with independent rows, such that for all $t \in [c], i \in [m]$, the Y_t and W_t equal to $\frac{1}{\sqrt{cp_i}}A_i$ and $\frac{1}{\sqrt{cp_i}}B_i$ respectively with probability p_i. Then, with probability no less than 0.7, for $Z = \arg\min_X ||Y - WX||_F$, we have

$$\begin{aligned} ||A - BZ||_F \leq& (1+\epsilon)||A - BZ_{opt}||_F \\ =& (1+\epsilon)\min_X ||A - BX||_F. \end{aligned} \tag{10}$$

Expected(c) sampling scheme from [3] provides a better asymptotic bound $O(k\log k\epsilon^{-2})$ on the number of required samples to achieve inequality (10). However, the constant factor of the bound is less obvious. Now we present the theoretical guarantee for Algorithm 1.

Theorem 2. *Assuming $d_1, d_2 \leq \min(m, n)$, and let $A \in \mathbb{R}^{m\times n}, C \in \mathbb{R}^{m\times d_1}$, $R \in \mathbb{R}^{d_2\times n}$ be three matrices, and $\epsilon < 1$ be any positive constant. Let Z be the output of Algorithm 1 with the above inputs, then with probability no less than 0.7, we have*

$$||A - CZR||_F \leq (1+\epsilon)\min_X ||A - CXR||_F. \tag{11}$$

Proof. If C(or R) is not a full rank matrix, we can locate and discard the extra columns(or rows) by performing algorithms such as rank-reveling QR factorization, and this can be done without losing precision in reconstructing A. Therefore, without loss of generality, assume that C and R be full rank matrices, and admit SVD decomposition $C = U_C\Sigma_C V_C^T = U_C S_C$, and $R = U_R\Sigma_R V_R^T = S_R V_R^T$. For simplicity, we name U_C and V_R as U and V, respectively. Therefore,

$$||A - U\hat{Z}V^T||_F = ||A - CZR||_F, \tag{12}$$

where $\hat{Z} = S_C Z S_R$.

The element on the i-th row and j-th column of UXV^T is

$$\left(UXV^T\right)_{i,j} = \sum_{a=1}^{d_1}\sum_{b=1}^{d_2} U_{i,a}X_{a,b}V_{j,b}, \tag{13}$$

and is equivalent to the inner product of $U_i \otimes V_j$ and $\vec{X}$. Here $\vec{M}$ denotes the vectorization of a matrix $M \in \mathbb{R}^{m\times n}$ such that $\vec{M} = [M_{1,1}, M_{1,2}, \ldots, M_{m,n}]^T$. Therefore

$$\begin{aligned} ||A - UXV^T||_F &= ||\vec{A} - \overrightarrow{UXV^T}|| && (14)\\ &= ||\vec{A} - (U\otimes V)\vec{X}||. && (15) \end{aligned}$$

Define $f : [m] \times [n] \to [mn]$ to be a bijection between the indices of a matrix and the indices of its vectorization, such that $f(i, j) = (i - 1)n + j$ for all $i \in [m]$, and $j \in [n]$. Since both U and V are orthogonal matrices, $U \otimes V$ is also orthogonal, and that

$$||(U\otimes V)_{f(i,j)}||_F^2 = \sum_{a=1}^{d_1}\sum_{b=1}^{d_2} U_{i,a}^2 V_{j,b}^2 \tag{16}$$

Draw independently with replacement $c = 3200d_1^2d_2^2\epsilon^{-2}$ random index pairs, $\{(i_t, j_t)|t \in [c]\}$, from probability distribution

$$\begin{aligned} p_{i,j} &= \text{Prob}\{i_t = i, j_t = j\} && (17)\\ &= \frac{||(U\otimes V)_{f(i,j)}||_F^2}{d_1 d_2}. && (18) \end{aligned}$$

Then construct sample vector $Y \in \mathbb{R}^c$ and matrix $W \in \mathbb{R}^{c\times d_1 d_2}$, such that for all $t \in [c]$,

$$Y_t = \frac{1}{\sqrt{cp_{i_t,j_t}}}A_{i_t,j_t} \tag{19}$$

and

$$W_t = \frac{1}{\sqrt{cp_{i_t,j_t}}}U_{i_t} \otimes V_{j_t}. \tag{20}$$

Solve the following regression problem

$$\bar{Z} = \arg\min_{\vec{X}} ||Y - W\vec{X}||, \tag{21}$$

By applying Theorem 1 with the A, B replaced with $\vec{A}$, $U\otimes V$ respectively, and setting $\beta = 1$, it can be easily shown that the $\bar{Z}$ computed above satisfies the following inequality with probability no less than 0.7,

$$||\vec{A} - (U\otimes V)\bar{Z}|| \leq (1+\epsilon)\min_{\vec{X}} ||\vec{A} - (U\otimes V)\vec{X}||. \tag{22}$$

Finally, reshape $\bar{Z}$ to $\hat{Z} \in \mathbb{R}^{d_1\times d_2}$, and let $Z = S_C^{-1}\hat{Z}S_R^{-1}$, then we have

$$\begin{aligned} ||A - CZR||_F &= ||A - U\hat{Z}V^T||_F && (23)\\ &= ||\vec{A} - \overrightarrow{U\hat{Z}V^T}|| && (24)\\ &= ||\vec{A} - (U\otimes V)\bar{Z}|| && (25)\\ &\leq (1+\epsilon)\min_{\vec{X}} ||\vec{A} - (U\otimes V)\vec{X}|| && (26)\\ &= (1+\epsilon)\min_X ||A - CXR||_F && (27) \end{aligned}$$

□

C. Relative Error Bound on $||A - CZR||_F$

In this subsection, we provide a near-optimal error bound analysis on the CUR decomposition Algorithm 1 produces, assuming sufficiently many columns and rows are sampled according to appropriate probability distributions.

Then we show that Algorithm 1, under conditions specified in Corollary 3.1, decomposes low-coherence input matrices near-optimally without accessing all entries and recovers unobserved entries in the process.

Before presenting the theorems, we first introduce the notations we use throughout this subsection. Given matrix

$A \in \mathbb{R}^{m\times n}$, and an integer k, $k \leq \min(m,n)$, and let U and V be the top k left and right singular vectors of A. We let

$$s_i = ||U_i||_F^2 \text{ for all } i \in [m] \tag{28}$$

denote the rank k row leverage scores of A, and similarly let

$$t_j = ||V_j||_F^2 \text{ for all } j \in [n] \tag{29}$$

denote the rank k column leverage scores of A. It is obvious that $\sum_{i=1}^m s_i = \sum_{j=1}^n = k$. Therefore, $\{s_i\}$ and $\{t_j\}$ naturally form two probability distributions $p_i = s_i/k, i \in [m]$ and $q_j = t_j/k, j \in [n]$.

We adopt the definition in [11] and let $\mu_r(A) = \max_i\{ms_i/k\}$, $\mu_c(A) = \max_j\{nt_j/k\}$, and $\mu(A) = \max\big(\mu_r(A), \mu_c(A)\big)$ denote the rank k row coherence, the rank k column coherence, and the rank k coherence, respectively. Notice that $r/m \leq \max_i\{s_i\} \leq 1$, and similarly $r/n \leq \max_j\{t_j\} \leq 1$. Therefore, $1 \leq \mu(A) \leq \max(m,n)$. We call A a low coherence matrix if $\mu(A)$ is a small constant, and $\mu(A) \ll \min(m/r, n/r)$.

Theorem 3. *Given $A \in \mathbb{R}^{m\times n}$, let $k \leq \min(m,n)$ be an integer, $\epsilon \in (0,1]$, and $c_0 = 3^2 \cdot 3200$ be constants. Assume that $d_1 \geq c_0 k^2\epsilon^{-2}$ columns are sampled with replacement according to probability distribution constructed with the column leverage scores of A, and construct $C \in \mathbb{R}^{m\times d_1}$ such that C consists of the sampled columns. Further assume that $d_2 \geq c_0 d_1^2\epsilon^{-2}$ rows are sampled with replacement according to probability distribution constructed with the row leverage scores, and construct $R \in \mathbb{R}^{d_2\times n}$, such that R consists of the sampled rows. Let Z be the output of Algorithm 1 with inputs A, C, R, and $\epsilon/8$, then with positive probability,*

$$||A - CZR||_F \leq (1+\epsilon)||A - A_k||_F.$$

Proof.

$$||A - CZR||_F \leq (1+\frac{\epsilon}{8})||A - CC^+AR^+R||_F \tag{30}$$
$$\leq (1+\frac{\epsilon}{8})(1+\frac{\epsilon}{3})||A - CC^+A||_F \tag{31}$$
$$\leq (1+\frac{\epsilon}{8})(1+\frac{\epsilon}{3})^2||A - A_k||_F \tag{32}$$
$$\leq (1+\epsilon)||A - A_k||_F \tag{33}$$

The first inequality is true due to Theorem 2, and the second and third inequalities are true due to Theorem 4 and Theorem 3 from [3] with $\epsilon/3$. All three inequalities have probability of success no less than 0.7, therefore taking union bound of the failure probability, inequality (33) holds with probability no less than 0.1. Notice that we can reduce the failure probability to δ by increasing the number of sampled columns, rows, and elements by $O(\log 1/\delta)$ times. □

Remark 1. *We can also achieve relative error CUR decomposition applying Algorithm 1 with C and R sampled using Expected(c) from [3] or the sampling scheme provided in [1]. The aforementioned two sampling schemes provide superior asymptotic bounds on the required number of sampled columns/rows($d_1 = O(k\log k\epsilon^{-2}), d_2 = O(d_1 \log d_1\epsilon^{-2})$ and $d_1, d_2 = O(k\epsilon^{-1})$, respectively). However, the constant factors on their bounds are less obvious.*

Corollary 3.1. *Given $A \in \mathbb{R}^{m\times n}$, let $k \leq \min(m,n)$ be an integer, and let $\epsilon \in (0,1]$ and $c_0 = 3^2 \cdot 3200$ be constants. Assume that the rank k coherence of A, $\mu(A) = \beta$, and that $d_1 \geq c_0 k^2\beta\epsilon^{-2}$ columns are sampled with replacement uniformly, and construct $C \in \mathbb{R}^{m\times d_1}$, such that C consists of the sampled columns. Further assume that $d_2 \geq c_0 d_1^2\epsilon^{-2}$ rows are sampled with replacement according to probability distribution constructed with the row leverage scores, and construct $R \in \mathbb{R}^{d_2\times n}$, such that R consists of the sampled rows. Let Z be the output of Algorithm 1 with inputs A, C, R, and $\epsilon/8$, then with positive probability,*

$$||A - CZR||_F \leq (1+\epsilon)||A - A_k||_F.$$

Remark 2. *If $\mu(A) = \beta$ is a small constant, then by setting the column sampling probability distribution to uniform we have loss of accuracy by at most $1/\beta$, i.e., $p_j = 1/n \geq ||V_j||_F^2/k\beta$ for all $j \in [n]$, and this can be compensated by sampling β times more columns. In the case where the columns and rows are sampled independently, and given $O(k^2\beta\epsilon^{-2})$ columns and rows are sampled uniformly with replacement, the error bound deteriorate to $(2+\epsilon)||A - A_k||_F$.*

D. Algorithm Complexity

In this section, we confirm that given $m, n >> d_1, d_2$, Algorithm 1 achieves sublinear complexity.

In the sampling stage, the sampling probability distribution $p_{i,j}$ should be computed implicitly, otherwise storing $p_{i,j}$ would already require mn space, exceeding the claimed sublinear complexity. Fortunately,

$$\text{Prob}\big\{i_t = i, j_t = j\big\} = \frac{||(U\otimes V)_{f(i,j)}||_F^2}{d_1 d_2} \tag{34}$$
$$= \frac{||U_i||_F^2}{d_1}\frac{||V_j||_F^2}{d_2} \tag{35}$$
$$= \text{Prob}\big\{i_t = i\big\}\text{Prob}\big\{j_t = j\big\}. \tag{36}$$

In other words, in the sampling stage, we can simply sample the row(column) index first, and then independently sample the other. Therefore, the dominating computational cost, $O(md_1^2 + nd_2^2)$, will be the cost for computing top singular vectors, which can be achieved through QR(or SVD) factorization of the input matrices C and R.

Let $c = O(d_1^2 d_2^2\epsilon^{-2})$ denote the number of samples required. The computational cost for constructing the down sampled problem $\min_X ||Y - WX||$ is $O(cd_1d_2)$, and this problem can be solved in closed form as W^+Y, whose cost is dominated by the cost, $O(cd_1^2d_2^2)$, of computing the pseudo-inverse of W. In conclusion, the complexity of Algorithm 1 is $O(md_1^2 + nd_2^2 + d_1^4d_2^4\epsilon^{-2})$.

V. Numerical Experiments

To demonstrate the empirical applicability of the proposed algorithm, we evaluate it on six large-scale real-world data matrices, some of which can contain over one billion values.

We implement the proposed CUR algorithm as well as the state-of-the-art CUR algorithms [3], [11] for comparisons. All tests are programmed using Python with libraries NumPy [8] and SciPy [6]. SciPy sparse matrix modules are used to handle those sparse input matrices. All the experiments are run on a PC with Intel I7 3.5GHz CPU, 16GB RAM, and Windows operating system.

A. CUR Matrix Approximation on Low-Coherence Matrices

In this subsection, we present the experimental results of the proposed algorithm on four benchmark data matrices for CUR matrix decomposition, which are widely used in previous work [9], [11].

The Enron Emails ($39,861 \times 28,102$), Dexter ($20,000 \times 2,600$), and Farm Ads ($54,877 \times 4,143$) are textual data where in their matrix form, each row associates with one document, and each column associates with one word, i.e., the element on the i-th row j-th column is the number of occurrence of word j in document i. Gisette ($13,500 \times 5,000$) data consists of hand-written digits. In its matrix form, each row corresponds to one written digit, and each column corresponds to one feature.

For each input data $A \in \mathbb{R}^{m\times n}$, we sample d_1 columns and d_2 rows uniformly. Let $C \in \mathbb{R}^{m\times d_1}$ be the matrix that consists of the sampled columns, and let $R \in \mathbb{R}^{d_2\times n}$. Then we compute $Z \in \mathbb{R}^{d_1\times d_2}$ as the return value of Algorithm 1 with inputs A, C, R, and c (i.e., number of samples). We compute the relative error of Z as

$$\textit{relative error} = \frac{\|A - CZR\|_F}{\|A - CZ_{opt}R\|_F} \tag{37}$$

where,

$$Z_{opt} = argmin_X \|A - CXR\|_F = C^+AR^+, \tag{38}$$

for performance evaluation.

For comparison, we also include the relative error of Z_+, where

$$Z_+ = \arg\min_X \sum_{(i,j)\in S_+} (A_{i,j} - C_iXR^j)^2, \tag{39}$$

and S_+ is a subset of c matrix entries sampled uniformly without replacement. This is essentially the output of the CUR+ algorithm [11] applied with the same C, R, and c. The key difference between CUR+ algorithm and Algorithm 1 is that in Algorithm 1, Z is computed with equation (1), where the summands are scaled, and S is a list of Ω matrix indices sampled independently with replacement according to a carefully constructed probability distribution.

In order to have comparable results, we follow the same experiment setting described in [11] by letting $d_1 = ar$, $d_2 = ad_1$, and $c = mnr^2/\mathrm{nnz(A)}$, where $\mathrm{nnz(A)}$ represents the number of nonzero elements in A. We let $r = 10$, and $a = 1, 2, 3, 4, 5$. We run each test 10 times and report the mean relative error of Z and Z_+.

In this experiment, the relative errors produced by Algorithm 1 equal to approximately 1.0 consistently, indicating the output Z accurately approximates C^+AR^+ using only a small fraction of the entries in A. We notice that relative errors for both algorithms spike in tests on the Gisette Data with $a = 2$. This is most likely caused by $c \approx d_1d_2$, which may lead to a close to square down sampled regression problem that has a larger condition number. As a increases from 1 to 5, the relative error increases for both outputs. This is because the number of rows and columns sampled increases substantially, but the number of sampled elements stays fixed, making it harder to recover C^+AR^+ . However we observe that the relative error of Algorithm 1 behaves rather stable and only deteriorates slightly.

B. CUR with Leverage Score Sampling

In this section, we confirm that Algorithm 1 can improve the approximation level of the randomized CUR algorithm using leverage score sampling. We perform experiments on the Jester and RCV1-v2 matrices used in [3]:

For both data matrices, we compute their rank 5 CUR approximation using the leverage score sampling for factor C and R, and Algorithm 1 for computing factor U. For comparison, we also compute the factor U as the pseudo-inverse of W, where W is the sub-block obtained by intersecting C and R.

Figure 2 displays the leading singular spectrum of both test matrices as well as the relative approximation errors for both Algorithm 1 and the CUR algorithm in [3] with number of sampled columns c ranged from 5 to 25. The corresponding number of sampled rows is set to be $2c$, and the number of sampled entries is set to be 4 times the size of U. The test runs 3 times for each value of c.

The Jester matrix is a dense matrix of size $14,116 \times 100$ with entry values representing user ratings between ± 10.0. Its best rank-5 approximation A_5 is capable of capturing 81% of the matrix Frobenius norm. Using 5 columns and 10 rows, the algorithm developed in [3] produces CUR approximation with relative error about 1.5, and the error steadily decreases to about 1.3 as the number of columns and rows are increased to 25 and 50. In comparison, Algorithm 1 that uses the exactly same set of columns and rows constantly produces better CUR approximations, with relative error decreased to about 1.1 in the end.

The RCV1v2 matrix is a sparse $47,236 \times 23,149$ matrix with 0.16% of its entries being nonzero. The rank-5 relative approximation errors are quite close to 1 even for $c = r = 5$, and increasing c and r does not seem to further improve the approximation accuracy. Compared to the baseline CUR algorithm, Algorithm 1 has more stable performance and constantly produces lower relative error.

VI. Conclusion

In this paper, we propose a novel randomized sublinear-time algorithm that provides approximately optimal solution to the double-sided least squares problem with high probability. We present theoretical results that guarantee the solution of our method will be close to the optimal low-rank approximation with high probability of success.

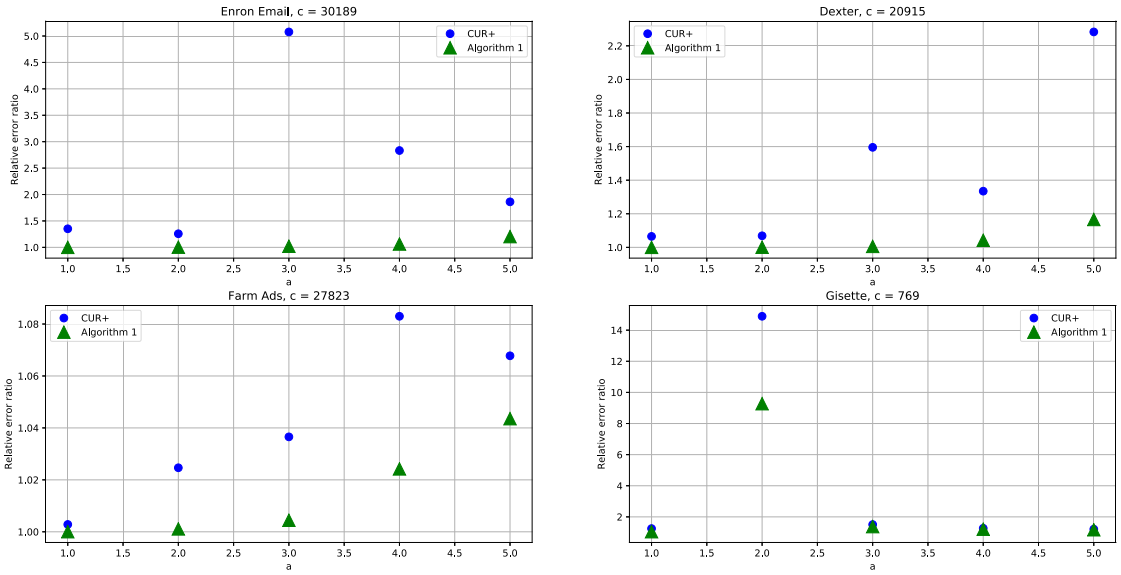

Fig. 1: Relative error produced by Algorithm 1 (this work) and the CUR+ algorithm in [11] on the Enron Email $(39,861 \times 28,102)$, Dexter $(20,000 \times 2,600)$, FarmAds $(54,877 \times 4,143)$, and Gisette $(13,500 \times 5,000)$ matrices, with $a = 1, 2, ..., 5$.

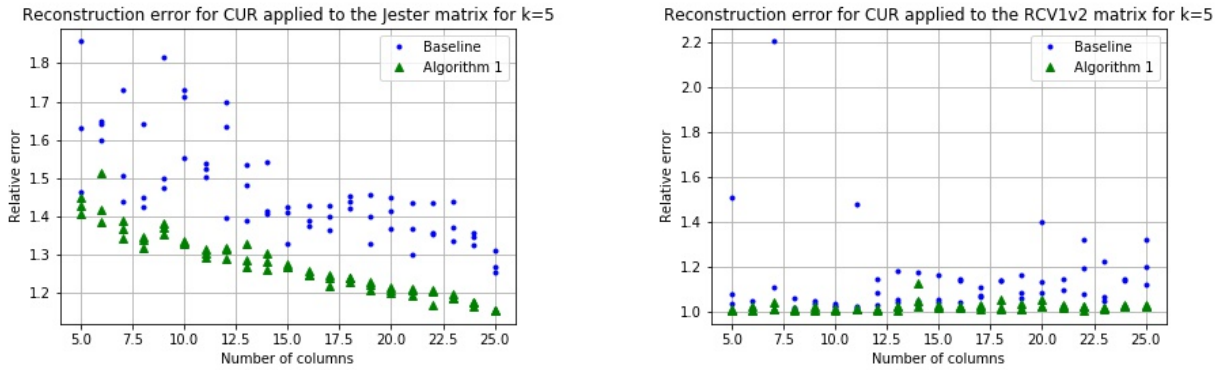

Fig. 2: Top left: All singular values of the $14,116 \times 100$ Jester matrix. Top right: Top 25 singular values of the $47,236 \times 23,149$ RCV1v2 matrix. Bottom: Relative error of CUR approximations produced by the CUR algorithm in [3] and Algorithm 1 using exact leverage score as sampling probability.

REFERENCES

[1] Christos Boutsidis and David P Woodruff. Optimal cur matrix decompositions. *SIAM Journal on Computing*, 46(2):543–589, 2017.

[2] Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast monte carlo algorithms for matrices iii: Computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 36(1):184–206, 2006.

[3] Petros Drineas, Michael W Mahoney, and Shan Muthukrishnan. Relative-error cur matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, 2008.

[4] Sergei A Goreinov, Eugene E Tyrtyshnikov, and Nickolai L Zamarashkin. A theory of pseudoskeleton approximations. *Linear algebra and its applications*, 261(1-3):1–21, 1997.

[5] Sergei A Goreinov, Nikolai Leonidovich Zamarashkin, and Evgenii Evgen'evich Tyrtyshnikov. Pseudo-skeleton approximations by matrices of maximal volume. *Mathematical Notes*, 62(4):515–519, 1997.

[6] Eric Jones, Travis Oliphant, and Pearu Peterson. {SciPy}: Open source scientific tools for {Python}, 2014.

[7] Eugene Tyrtyshnikov. Incomplete cross approximation in the mosaic-skeleton method. *Computing*, 64(4):367–380, 2000.

[8] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22, 2011.

[9] Shusen Wang and Zhihua Zhang. Improving cur matrix decomposition and the nyström approximation via adaptive sampling. *The Journal of Machine Learning Research*, 14(1):2729–2769, 2013.

[10] David P Woodruff et al. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.

[11] Miao Xu, Rong Jin, and Zhi-Hua Zhou. Cur algorithm for partially observed matrices. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, pages 1412–1421. JMLR. org, 2015.